消息队列 CMQ

SDK 文档

产品文档

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2019 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

SDK 文档
- SDK 更新日志
- HTTP SDK
- TCP SDK

版权所有：腾讯云计算（北京）有限责任公司

SDK 文档
SDK 更新日志

最近更新时间：2021-12-29 16:02:56

CMQ HTTP SDK 1.0.7（2020-03-25）

- 新版的 artifactId 改为"cmq-http-client"。
- 修复异常信息被吞的 Bug。
- 移除掉了未使用 Client，同时修复了 issue#6、#7。

代码仓库地址 >>

CMQ TCP SDK 1.1.1（2020-02-12）

新特性

- 添加管理流相关接口，如创建队列、创建订阅等。
- 补全相关单元测试，默认构建时不开启。
- pom 文件兼容 maven 3.0+。

Bug 修复

- 修复事务消息异常被吞问题。
- 修复配置中超时时间对 TCP 无效的问题。
- 修复并发场景中出现“重复鉴权”的异常。

代码仓库地址 >>

CMQ HTTP SDK 1.0.6（2019-11-12）

新特性

- SDK 的配置对象化，增加超时时间、是否打印慢操作日志等参数，详情见 CmqConfig 对象。
- 新接收消息、发送接口性能提升6倍。
- 新增自动慢操作、异常日志打印（需用户配置日志文件）。
- 新发送接口增加 requestId 的返回，方便客户问题跟进。

Bug 修复

- 修复偶发 JSON 序列化异常问题。
- 其他代码规范问题的修复。
- 日志配置操作。

版权所有：腾讯云计算（北京）有限责任公司

- SDK 已经使用 slf4j 打印日志，但没有具体的日志配置文件，用户在 classpath 中添加 log4j 或 logback 等日志框架的配置文件即可。

使用建议

- 使用新的（batch）receiveMessage 接口接收消息（长轮询等待时间由队列的设置决定）。
- 使用新的（batch）send 接口发送消息（可以返回 requestId）。

代码仓库地址 >>

CMQ SDK 1.05（2019-06-28）

CMQ 支持基于 TCP 协议的 SDK 调用，使用更少的计算资源、更安全的客户端线程、更高效的传输效率、更优的使用体验、更多样的特性支持。

CMQ SDK 1.0.4（2017-04-07）

- CMQ SDK 支持 sha256 签名，您可在初始化 Account 中调用 setSignMethod（具体方法名可参考代码说明）设置签名方法。
- 修复已知 Bug。

CMQ SDK 1.0.3（2017-03-13）

CMQ 新增特性：消息回溯、消息延时、订阅路由功能。

CMQ SDK 1.0.2（2016-12-01）

- SDK 支持主题和订阅模式。
- C++ SDK 改用 cmake 管理项目。
- Java SDK 改用 maven 管理项目。

CMQ SDK 1.0.1（2016-10-12）

- 优化客户端超时体验。
- 修复 PHP SDK 鉴权失败 Bug。
- 修复 PHP 发送消息 Bug。

CMQ SDK 1.0.0（2016-09-07）

- 同时上线 C++、Java、Python、PHP 四个语言版本。
- SDK 封装了发送消息、接收消息、删除消息等操作接口。

版权所有：腾讯云计算（北京）有限责任公司

HTTP SDK

最近更新时间：2020-11-20 15:40:11

概述

腾讯云消息队列 CMQ 目前支持 Java、Python、PHP 及 C++ SDK ，后续会支持更多语言。欢迎广大开发者根据 API 说明开发更多语言版本的 SDK。
由于分配资源和释放资源需要1s左右的时间，当前消息队列 SDK 在创建及删除队列/主题时会有1s 延迟，建议在程序中增加创建和删除的时间间隔保障调用成功。

注意：

- 为了保障数据传输的安全，为您提供更加可靠的服务，腾讯云团队将在2019年1月31日23:59:59停止对主题及队列的公网 HTTP 方式的访问，建议您将公网接口请求域名改为 HTTPS。
- 修改公网接口请求域名的方式如下：
  判断接入域名中使用的协议是否为 HTTP，如果有，将其替换为 HTTPS 即可。
  例如： `endpoint=http://cmq-topic-gz.api.qcloud.com` 该接入方式为 HTTP 协议访问广州主题公网域名，修改为： `endpoint=https://cmq-topic-gz.api.qcloud.com` 。

下载地址

不同语言版本 SDK 的 GitHub 地址如下：

- Java SDK
- Python SDK（默认为 Python2 SDK，您可切换至 Python3 分支中查看 Python3 SDK ）
- PHP SDK
- C++ SDK

注意事项

使用 SDK 前至少要获取 SecretId、 SecretKey 和 endpoint（即请求发到哪个地域，走内网还是外网）。endpoint 说明如下。

队列模型

请参照下面说明将域名中的 {$region} 替换成相应地域：

- 外网接口请求域名： `https://cmq-queue-{$region}.api.qcloud.com`
- 内网接口请求域名： `http://cmq-queue-{$region}.api.tencentyun.com`

">

主题模型

请参照下面说明将域名中的 {$region} 替换成相应地域：

- 外网接口请求域名： `https://cmq-topic-{$region}.api.qcloud.com`
- 内网接口请求域名： `http://cmq-topic-{$region}.api.tencentyun.com`

如果业务进程也部署在腾讯云的 CVM 子机上，强烈建议使用同地域的内网 endpoint。例如：在腾讯云北京地域的 CVM 子机，则建议您使用 `http://cmq-queue-bj.api.tencentyun.com` 。原因如下：

- 同地域内网时延更低。
- 目前消息队列对于公网下行流量是要收取流量费用的，用内网可以节省这部分的费用。

说明

{$region}需用具体地域替换：gz（广州），sh（上海），bj（北京），shjr（上海金融），szjr（深圳金融），hk（中国香港），cd（成都），ca(北美)，usw（美西），sg（新加坡）。公共参数中的 region 值要与域名的 region 值保持一致，如果出现不一致的情况，以域名的 region 值为准，将请求发往域名 region 所指定的地域。

Demo工程使用

准备 Demo 环境

1. **安装 IDE**
   您可以安装 IntelliJ IDEA 或者 Eclipse，本文以 IntelliJ IDEA 为例进行说明。
   请在 下载 IntelliJ IDEA Ultimate 版本，并参考 IntelliJ IDEA 说明进行安装。
2. **下载 Demo 工程**
   请在 下载 CMQ-HTTP 的 Demo 工程 到本地，解压后即可看到本地新增的 cmq-java-sdk-master 文件夹。

配置 Demo 工程

1. **创建资源**
   您需要在控制台创建所需消息队列资源，包括 CMQ 队列名、SecretID、SecretKey。
   具体创建过程请参考 队列模型快速入门 和 主题模型快速入门。
2. **导入 Demo 工程文件**
   在 IDEA 的开机界面打开文件夹。

打开文件夹后，Demo 工程文件存于 `/src/main/java/com/qcloud/cmq/example` 文件夹下。

3. **配置 Demo 参数**

修改文件请求地址、密钥对等。以 Producer 为例，配置如下：

```
String secretId="获取的SecretID";
String secretKey="获取的SecretKey";
String endpoint = "https://cmq-topic-{$region}.api.qcloud.com";
String queueName = "test";
```

选择“新建队列”和“使用已有队列”：

```
//创建新队列并设置属性
QueueMeta meta = new QueueMeta();
meta.pollingWaitSeconds = 10;
meta.visibilityTimeout = 10;
meta.maxMsgSize = 1048576;
meta.msgRetentionSeconds = 345600;
Queue queue = account.createQueue(queueName,meta);
```

```
//使用控制台已有队列
Queue queue = account.getQueue(queueName);
```

本 Demo 使用的是当前账号已有队列，如需选择新建队列，修改 queueName 为将要创建的队列名，然后找到新建队列相关代码取消注释。

注释代码均为常用操作，“删除”操作需谨慎使用，其他类中的配置参考 Producer 类。

运行 Demo

使用队列模型收发消息

先运行 Producer 类发送消息，再运行 Consumer 类接受消息。

发送消息代码示例：

```
String msg = "hello!";
String msgId = queue.sendMessage(msg);
System.out.println("==> send success! msg_id:" + msgId);
```

接收消息代码示例：

```
Message msg = queue.receiveMessage(10);
```

使用主题模型收发消息

运行 TopicDemo 类，主题模型请求域名参考 主题模型请求域名。

发布消息示例：

```
String msg = "hello!";
String msgId = topic.publishMessage(msg);
```

处理消息示例：

```
String queueName = "test";
String subscriptionName = "sub-test";
String Endpoint = queueName;
String Protocol = "queue";
account.createSubscribe(topicName,subscriptionName, Endpoint, Protocol);
```

创建订阅者时填写一个队列，用队列处理消息。

版权所有：腾讯云计算（北京）有限责任公司

TCP SDK

最近更新时间：2020-09-10 14:54:27

CMQ 目前已开放基于 TCP 协议的 SDK 调用，支持普通消息、事务消息、延迟消息、异步消息的收发功能。其中，事务消息特性仅通过 TCP 方式实现。
TCP 协议目前支持公网访问以及同地域的私有网络的 CVM 内网访问方式，暂时不支持基础网络的内网访问方式。
本文主要介绍 TCP SDK 使用方式，提供 Demo 工程的安装、下载、配置及运行示例，帮助工程师快速搭建 CMQ 测试工程。

TCP 协议优势

- **更少的计算资源**
  HTTP 针对请求鉴权，每次请求都需要签名；TCP 针对链接鉴权，只需要建立连接的时候对链接鉴权，节约客户端计算资源。
- **更安全的客户端线程**
  TCP 客户端线程安全，多个线程可使用相同的链接，节省链接资源。
- **更高效的传输效率**
  TCP 传输提高有效数据占比，在相同客户端下，拥有更高的吞吐量和 QPS，相比 HTTP 具有更高效的传输效率。
- **更优的使用体验**
  TCP 支持异步接口，支持回调。
- **更多样的特性支持**
  TCP 支持 CMQ 最新的分布式消息事务。

Demo工程使用

准备 Demo 环境

1. **安装 IDE**
   您可以安装 IntelliJ IDEA 或者 Eclipse，本文以 IntelliJ IDEA 为例进行说明。
   请在 下载 IntelliJ IDEA Ultimate 版本，并参考 IntelliJ IDEA 说明进行安装。
2. **下载 Demo 工程**
   请在 下载 CMQ 的 Demo 工程 到本地，解压后即可看到本地新增的 cmq-java-tcp-sdk-master 文件夹。

配置 Demo 工程

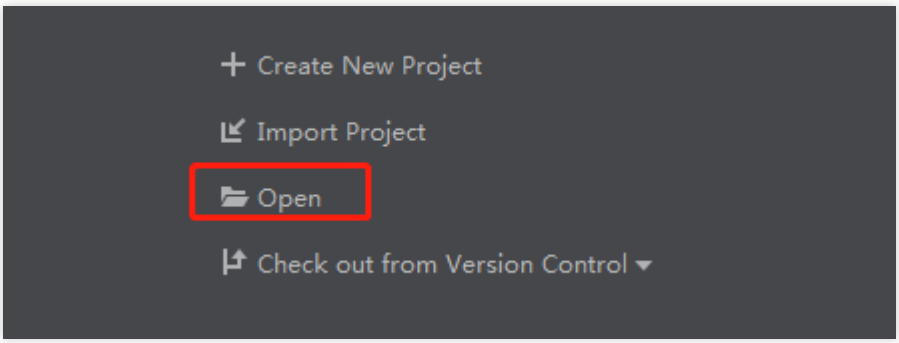
1. **创建资源**

   您需要在控制台创建所需消息队列资源，包括 CMQ 队列名、SecretID、SecretKey。

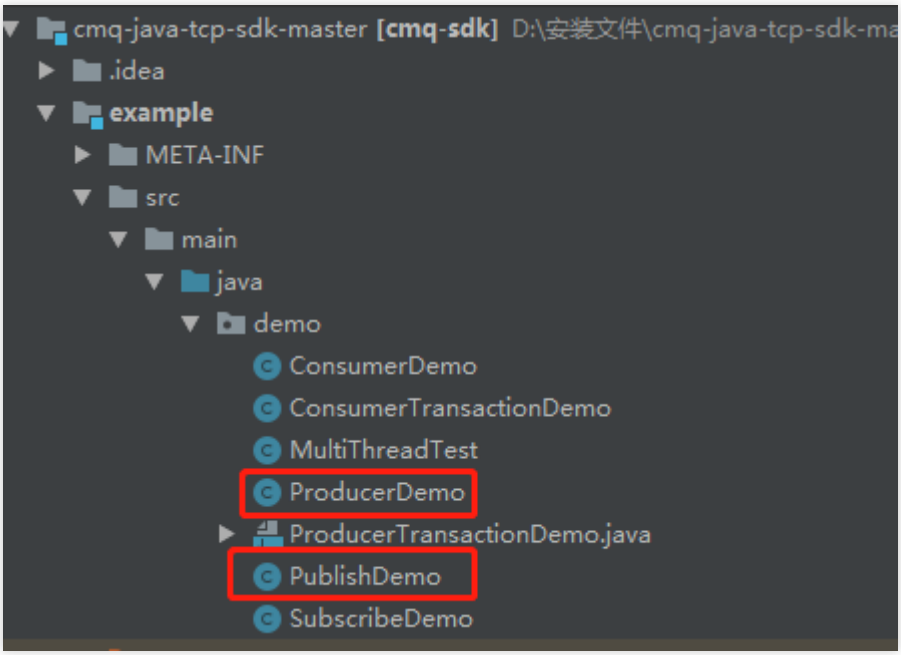
   具体创建过程请参考 队列模型快速入门 和 主题模型快速入门。

2. **导入 Demo 工程文件**

   在 IDEA 的开机界面打开文件夹。



   打开文件夹后，文件层级关系如下，Demo 工程文件存于 Demo 文件夹下。



3. **配置 Demo 参数**

   修改文件 NameServer 地址、密钥对及消息队列名，NameServer 地址请参考 NameServer 对照表。

   以 ProducerDemo 为例，配置如下：

```
producer.setNameServerAddress("对应的NameSever");
producer.setSecretID("获取的SecretID")
producer.setSecretKey("获取的SecretKey")
String queue = "创建的队列名"
```

具体图示如下：

```
public class ProducerDemo {
    public static void main(String args[]) {

        Producer producer = new Producer();

        producer.setNameServerAddress("http://cmq-nameserver-bj.tencentcloudapi.com");

        producer.setSecretId("AKIDX1LbH2DNo8MWmOdx[redacted]DVWc");

        producer.setSecretKey("F2jJjAWr5A5v8D8sT[redacted]");

        producer.setSignMethod(ClientConfig.SIGN_METHOD_SHA256);

        producer.setRetryTimesWhenSendFailed(3);

        producer.setRequestTimeoutMS(8000);


        String queue = "subuin";
        try {
```

运行 Demo

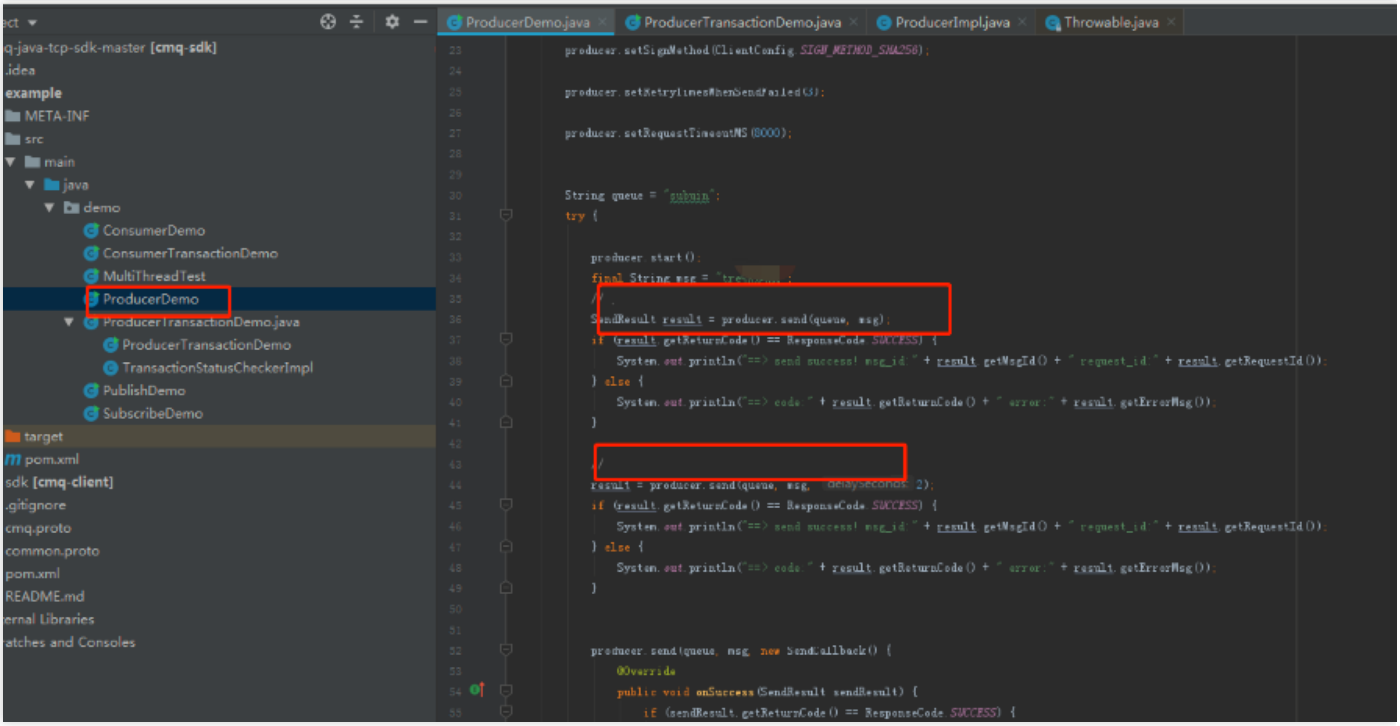
使用队列模型收发消息

1. 配置 Demo 参数。
2. 执行文件 ProducerDemo，成功后显示日志如下：

```
2019-06-26 17:26:32,448 INFO  NettyClient.java(666) : createChannel: begin to connect remote host[140.143.52.21:12000] asynchronously
2019-06-26 17:26:32,470 INFO  NettyClient.java(760) : NETTY CLIENT PIPELINE: CONNECT  UNKNOWN => 140.143.52.21:12000
==> send success! msg_id:4222124650659902 request_id:1
==> send success! msg_id:4222124650659903 request_id:3
==> send success! msg_id:4222124650659904 request_id:4
==> send success! msg_id:[4222124650659905, 4222124650659906] request_id:3
==> send success! msg_id:[4222124650659907, 4222124650659908] request_id:6
2019-06-26 17:26:37,890 INFO  NettyClient.java(573) : closeChannel: begin close the channel[140.143.52.21:12000] Found: true
2019-06-26 17:26:37,890 INFO  NettyClient.java(586) : closeChannel: the channel[140.143.52.21:12000] was removed from channel table
2019-06-26 17:26:37,891 INFO  NettyClient.java(775) : NETTY CLIENT PIPELINE: CLOSE 140.143.52.21:12000
2019-06-26 17:26:37,892 INFO  MQClientInstance.java(108) : the client instance [10.66.130.82@21656] shutdown OK
```

版权所有：腾讯云计算（北京）有限责任公司

ProducerDemo 支持普通消息、延时消息、异步消息的发送。



3. 执行文件 ConsumerDemo，可接收消息。

使用主题模型收发消息

1. 运行 PublishDemo 类以主题模型进行消息发送。
2. 运行 SubscriberDemo 类以主题模式进行消息接收。

收发事务消息

1. 运行 ProducerTransactionDemo 类进行事务消息发送。
2. 运行 SubscriberTransactionDemo 类进行事务消息接收。

">

Nameserver 对照表

地区	公网地址	VPC 地址
印度	http://cmq-nameserver-in.tencentcloudapi.com	http://cmq-nameserver-vpc-in.api.tencentyun.com
北京	http://cmq-nameserver-bj.tencentcloudapi.com	http://cmq-nameserver-vpc-bj.api.tencentyun.com

版权所有：腾讯云计算（北京）有限责任公司

上海	http://cmq-nameserver-sh.tencentcloudapi.com	http://cmq-nameserver-vpc-sh.api.tencentyun.com
广州	http://cmq-nameserver-gz.tencentcloudapi.com	http://cmq-nameserver-vpc-gz.api.tencentyun.com
北美	http://cmq-nameserver-ca.tencentcloudapi.com	http://cmq-nameserver-vpc-ca.api.tencentyun.com
成都	http://cmq-nameserver-cd.tencentcloudapi.com	http://cmq-nameserver-vpc-cd.api.tencentyun.com
重庆	http://cmq-nameserver-cq.tencentcloudapi.com	http://cmq-nameserver-vpc-cq.api.tencentyun.com
中国香港	http://cmq-nameserver-hk.tencentcloudapi.com	http://cmq-nameserver-vpc-hk.api.tencentyun.com
韩国	http://cmq-nameserver-kr.tencentcloudapi.com	http://cmq-nameserver-vpc-kr.api.tencentyun.com
俄罗斯	http://cmq-nameserver-ru.tencentcloudapi.com	http://cmq-nameserver-vpc-ru.api.tencentyun.com
新加坡	http://cmq-nameserver-sg.tencentcloudapi.com	http://cmq-nameserver-vpc-sg.api.tencentyun.com
泰国	http://cmq-nameserver-th.tencentcloudapi.com	http://cmq-nameserver-vpc-th.api.tencentyun.com
美东	http://cmq-nameserver-use.tencentcloudapi.com	http://cmq-nameserver-vpc-use.api.tencentyun.com
美西	http://cmq-nameserver-usw.tencentcloudapi.com	http://cmq-nameserver-vpc-usw.api.tencentyun.com

版权所有：腾讯云计算（北京）有限责任公司